REMARKS/ARGUMENTS

Reconsideration of this application is respectfully requested.

The continued rejection of all claims 23-42 under 35 U.S.C. §102 as allegedly anticipated by Hacherl '571 is again respectfully traversed.

Applicants' earlier remarks regarding Hacherl are still believed to be applicable and, so as to not unduly burden the record, instead of repeating such remarks, the Examiner is respectfully requested to reconsider arguments already of record with respect to Hacherl.

The Examiner is thanked for providing a "response to arguments" section at pages 2-4 of the office action.

With respect to what the Examiner has designated as "point A", the Examiner asserts that Hacherl does, in fact, teach the claimed data objects because, the Examiner asserts, the Hacherl data objects do contain "data and dynamic elements comprising attributes and methods". With respect, this assertion is clearly erroneous.

The Examiner's first attempt to support this assertion relies upon Hacherl at 1:10-15. However, this text merely comprises the definition of the "technical field" to which Hacherl is directed. The fact that Hacherl may relate generally to distributed computer systems, even systems which dynamically allocate exclusive authority for performing network-wide tasks among different computers in a network, does not have anything at all to do with whether or not such a distributed computer system employs any data objects that contain not only data, but also dynamic elements comprising attributes and methods. Indeed, as will be explained in more detail below, Hacherl teaches the use of data objects which contain only data – and <u>not</u> dynamic elements comprising attributes and methods.

The Examiner next cites Hacherl at 3:40-43 to support this assertion. However, the text found at this portion of Hacherl simply describes the computer environment depicted in Fig. 1 and states that, in general, <u>program modules</u> include routines, programs, objects, components, data structures and the like that perform particular tasks or implement particular abstract data types. As those having skill in the art will recognize, the fact that the Hacherl system might, somewhere, include "program modules" has nothing whatever to do with whether the Hacherl system passes data objects between processors in a network which contain not only data, but also dynamic elements comprising attributes and methods. Indeed, as will be demonstrated more completely below, Hacherl does not have such teaching. That is, Hacherl's "program modules" do not appear to be part of objects being exchanged over the network.

The Examiner next attempts to find support for this assertion in Hacherl at 6:34-41. However, this passage merely describes the directory service 80 which provides a name space in which the name of an object in the directory can be resolved to the object itself. Indeed, this text explicitly explains that the Hacherl "object" is a distinct, named set of attributes that represents something concrete, such as a user, a printer, or an application. Hacherl goes on to explain that the "attributes" hold data describing the thing that is identified by the directory object. Attributes of a user might include the user's given name, surname and e-mail address. Clearly, there is no teaching here of anything approaching the applicants' claimed data object which contains not only data, but also dynamic elements comprising attributes and methods.

Finally, the Examiner attempts to find support for this assertion at Hacherl's claim 10 because claim 10 recites a method of transferring ownership of a master role from a current master server in a plurality of servers to a requesting server in the plurality of servers. Of course, merely passing the role of "master" between servers in a network again does not in any way teach (or even suggest) the passing of data objects between servers which include not only data, but also dynamic elements comprising attributes and methods. Merely signalling a change in status from slave to master does not constitute the use of data objects containing not only data, but also dynamic elements comprising attributes and methods.

Accordingly, when the Examiner's citations for support are actually examined, they do not support the assertion that Hacherl's data objects contain "data and dynamic elements comprising attributes and methods".

With respect to what the Examiner has designated "point B", the Examiner asserts that Hacherl teaches data objects including both dynamic elements and data <u>and</u> discloses <u>updating</u> data in such an object without changing the dynamic elements. Once again, this assertion is clearly erroneous.

As the sole support cited by the Examiner for this assertion, the Examiner refers to Hacherl at 7:55-62. However, this passage deals with a destination server updating its local copy of the agreement with the source's current update number so that it will not continuously fetch the same changes. Indeed, the cited passage goes on to <u>explicitly</u> teach <u>away</u> from merely <u>updating</u> a portion of a data object because this passage of Hacherl explicitly teaches that, in the exemplary Hacherl network, objects are <u>replicated</u> – and specifically contrasts the replication procedure with a "collection of changes to be applied..." This aspect of Hacherl will be discussed more completely below when the applicants' independent claim features are individually addressed in more detail.

As to what the Examiner has designated "point C", the Examiner asserts that Hacherl teaches "relevance" to networked games. Here, the Examiner relies upon the text found at 4:34-45 and/or 5:5-10. However, the cited text at col. 4 merely deals with standard input devices – and the text at col. 5 cited by the Examiner merely describes a network of servers/workstations, etc. Neither of the cited passages has any teaching whatsoever with respect to a networked game environment or maintenance of such environment.

Accordingly, the Examiner's attempt to rebut applicants' earlier submission self-evidently fails.

In an attempt to further illustrate why it is not conceivable that Hacherl anticipates any of applicants' claims, some of the more salient limitations of applicants' independent claims will now be individually discussed *vis-à-vis* the Hacherl teaching.

Hacherl does not disclose a first data object that is duplicated to each of the other terminals, and second data objects each of which is a duplicate of a data object on another terminal, as required by the applicants' independent claims.

The Examiner asserts that Hacherl discloses the claimed data objects because the claimed definition (a data object contains data and dynamic elements comprising attributes and methods), Hacherl's program modules (see 3:40-44) read onto these data objects. This is incorrect as explained above. Furthermore, Hacherl's program modules are not duplicated to other network terminals, as is clearly required by the applicants' claims.

Although Hacherl's "program modules may be located both in local and remote memory storage devices" (3:52-54), the specification as a whole (e.g., 4:67 to 5:2) makes it clear that "both" in this phrase means "either". There is no disclosure in Hacherl of replicating the program modules.

The Examiner additionally (and in contradiction of the Examiner's own argument) attempts to read the "first data object" of Hacherl's claim 1 onto the first data object of the applicants' independent claims, and the "second object" of Hacherl's claim 1 onto the applicants' claimed second data objects. However, Hacherl's objects, according to the specification at 6:36-39, are simply collections of attributes, i.e., data – and not containing methods or dynamic elements.

Thus, the Examiner is conflating Hacherl's program modules (which are not replicated) and the Hacherl objects (which are replicated, but are not "objects" within the scope of applicants' claims) to anticipate the duplicated data objects of the applicants' independent claims. Clearly, therefore, Hacherl does not disclose these claim limitations – indeed, Hacherl teaches away from the claimed structure.

The applicants' independent claims also require maintaining an environment for a networked game, displaying the environment on a display and, for at least one of the claimed data objects, generating an entity using methods and attributes in the claimed data object and displaying it in the environment.

Hacherl does not disclose anything like this. The Examiner notes that Hacherl teaches conventional input devices that could be used for a game, such as a joystick or game pad (see 4:34-45). However, the fact that these conventional input devices could be used does not mean that Hacherl teaches the claimed usage. The Examiner also notes that "program modules may be located in both local and remote memory storage devices". Of course, this is not relevant to the display of entities within a game environment.

The Examiner has provided no rationale or explanation as to why Hacherl is considered to anticipate the claimed game environment features. It is, therefore, clear that the Examiner has failed to established even a *prima facie* case of anticipation.

The applicants' independent claims also require periodically providing, over the network, an update of the data contained in the first data object and updating the data contained in the second objects in response to receiving updates over the network. It is emphasized in these claims that it is *a portion of the data* only that is updated, not the entire data object (i.e., not the method). Even if Hacherl's objects are considered *arguendo* to be the claimed data objects (which as discussed above they cannot), Hacherl does not update data in objects in this way.

When Hacherl's system duplicates the directory, it is entire objects that are duplicated. This is emphasized at 7:58-62:

"Thus the data transmitted in a replication procedure consists of updated objects, as opposed to a collection of changes to be applied, that are to be used in overwriting objects on the destination domain controller."

This makes it clear that Hacherl's objects are overwritten, not updated.

The Examiner has noted the Hacherl sentence:

"After a packet is successfully applied, the destination server will update its local copy of the agreement with the source's current update number."

This "replication agreement" is described by Hacherl at 7:20-26, and it is not itself an object. Thus, the sentence quoted by the Examiner is not relevant to the applicants' independent claims – except to the extent that it teaches away from the claimed invention.

Accordingly, the sole outstanding rejection of all claims as allegedly being anticipated by Hacherl '571 is clearly erroneous. The Examiner has failed to establish even a *prima facie* case of anticipation with respect to any single claim. Therefore, it is not necessary at this time to discuss additional deficiencies of Hacherl with respect to other aspects of the rejected claims. Suffice it to note that, as a matter of law, it is impossible to establish even a *prima facie* case of anticipation unless the cited single prior art reference teaches each and every feature of each and every rejected claim.

Accordingly, this entire application is now believed to be in allowable condition, and a formal notice to that effect is earnestly solicited.

Respectfully submitted,

NIXON & VANDERHYE P.C.

By: Larry S. Nixon
Larry S. Nixon
Reg. No. 25,640

LSN:lef

901 North Glebe Road, 11th Floor
Arlington, VA 22203-1808
Telephone: (703) 816-4000
Facsimile: (703) 816-4100